Screen Output – Numbers and Strings

Syntax

- Syntax refers to a high level programming language's vocabulary and grammar

- In order to successfully instruct a computer what to do using a high-level programming language you must follow the syntax of the language (vocabulary and grammar)

Recall: instructions in Python

print (“Game Over”)

- **print** - is the command – it does something
- “Game Over” - is an expression – it is something
- **print** “Game Over” is a statement
  - In programming a statement is a complete instruction

String Literals

- In the case of the instruction **print** “Game Over” , the expression is a string literal

- A string literal: a sequence of characters in quotes

For example:

Syntax

print (“text is in here 1”)

will produce the following in the message Screen:

text is in here 1

Notice:

- the string is duplicated literally ie. 1 space = 1 space, no space = no space; what you see is what you get.
- the quotes are not written
- numbers and non-letters can be part of the string

Formatting literal strings

- The text in a literal string can be manipulated using escape sequences
  - These allow you to put special characters into your strings
  - Composed of two characters: a ‘\’ followed by another character

Escape Sequences

\n	Newline. Moves cursor to beginning of next line
\t	Tab. Moves cursor forward one tab stop
\b	Backspace. Moves cursor back one space
\"	Double quote. Prints a double quote.
\'	Single quote. Prints a single quote.
\\	Backslash. Prints a Backslash.

An example

Code	Output
print (”Hithere 316”)	
print (”hi\tthere\nbob”)	
print (”abc\’def\’”)	
print (”The\\\\n\”is\”:”)	

Code	Output
print (”Hithere 316”)	Hithere 316
print (”hi\tthere\nbob”)	hi there bob
print (”abc\’def\’”)	abc'def'
print (”The\\\\n\”is\”:”)	The\\n"is":

Concatenating Strings

- Means joining two or more literal strings together
- Simply use a ‘+’ to join two or more strings
- E.g.
  - print (“This ” + “is ” + “so fun”)

Suppressing a NewLine

- To output two expressions from two successive print statements, use a comma:

- E.g.

```
print("This is ", end="")
print("so fun")
```

  - Will output: This is so fun

Repeating Strings

- Use a ‘*’ to indicate the number of times you want to repeat the expression

- E.g.
  - print (“Pie ”*5)
  - Will output:

    Pie Pie Pie Pie Pie

Displaying Numbers

Displaying Numbers

- A number has a mathematical numerical value

- Two types of numbers:
  - Integers – whole number
    - E.g. 1, 24, 50, 1000, etc.
  - Floats (floating point numbers) – decimals
    - E.g. 2.3, 54.2, 50.0

<u>Syntax</u>

print (3)

print (5.4)

(notice it is not a string so no quotations are needed)

Mathematical Operators

- Python follows all mathematical principles (including order of operations) and operators. For example:
  - + addition
  - - subtraction
  - * multiplication
  - / division
  - % modulus (find the remainder)
    - e.g. print 9 % 2 will output the remainder of the expression 9 % 2 which is 1.

- E.g.
  - print 2000 – 100 + 50
    - Will evaluate the expression 2000-100+50 and output 1950

Math Operations with Integers

Operator	Example	Evaluates to
*	7 * 3	21
/	7 / 3	2
%	7 % 3	1
+	7 + 3	10
-	7 – 3	4

Math Operations with Floats

Operator	Example	Evaluates to
*	7.0 * 3.0	21.0
/	7.0 / 3.0	2.333333333
%	7.0 % 3.0	1.0
+	7.0 + 3.0	10.0
-	7.0 – 3.0	4.0

Note:

- In Python, you must show all operations

- For example in math the expression 3(4+5) must be written in Python as 3*(4+5)

Display Numbers AND String Literals

Displaying a combination of information

To combine different types of info in one print statement - use a comma

Code	Output
print (“x”, 3)	x 3
print (“8*8=“,(8*8))	8*8= 64
print (5,”85”,24.2,”\n5*2”)	5 85 24.2 5*2